**INVENTORY MANAGEMENT:**

#include <iostream>

#include <vector>

#include <algorithm> // Include the algorithm header for find\_if

using namespace std;

// Structure to represent a product

struct Product {

int id;

string name;

double price;

int quantity;

};

// Function to add a new product to the inventory

void addProduct(vector<Product>& inventory, int id, const string& name, double price, int quantity) {

// Check if the product with the given ID already exists

for (const Product& product : inventory) {

if (product.id == id) {

cout << "Product with ID " << id << " already exists. Use a different ID.\n";

return;

}

}

// Create a new product and add it to the inventory

Product newProduct = {id, name, price, quantity};

inventory.push\_back(newProduct);

cout << "Product added successfully.\n";

}

void removeProduct(vector<Product>& inventory, int id) {

auto it = find\_if(inventory.begin(), inventory.end(), [id](const Product& product) {

return product.id == id;

});

if (it != inventory.end()) {

inventory.erase(it);

cout << "Product with ID " << id << " removed successfully.\n";

} else {

cout << "Product with ID " << id << " not found in the inventory.\n";

}

}

void displayInventory(const vector<Product>& inventory) {

cout << "Current Inventory:\n";

for (const Product& product : inventory) {

cout << "ID: " << product.id << ", Name: " << product.name << ", Price: " << product.price

<< ", Quantity: " << product.quantity << "\n";

}

}

int main() {

vector<Product> inventory;

addProduct(inventory, 1, "Laptop", 699.99, 5);

addProduct(inventory, 2, "Smartphone", 345.99, 10);

addProduct(inventory, 1, "Tablet", 299.99, 8);

displayInventory(inventory);

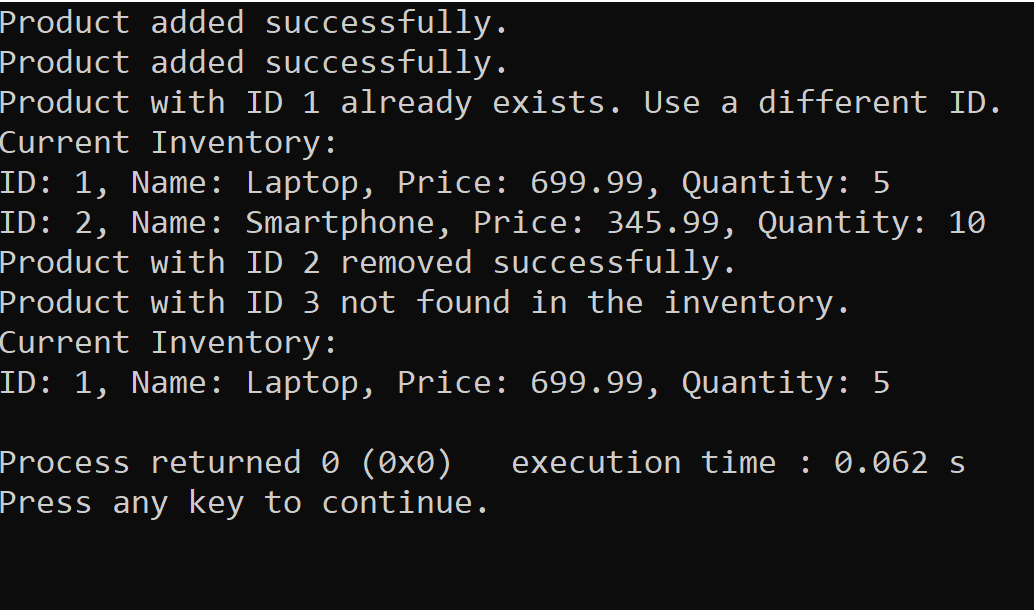
removeProduct(inventory, 2);

removeProduct(inventory, 3);

displayInventory(inventory);

}

**OUTPUT:**



**SORTING PERFORMANCE:**

#include <iostream>

#include <vector>

#include <algorithm>

#include <chrono>

int main() {

std::vector<int> arr(100000);

for (int i = 0; i < 100000; i++) {

arr[i] = 100000 - i;

}

auto start = std::chrono::high\_resolution\_clock::now();

std::sort(arr.begin(), arr.end());

auto end = std::chrono::high\_resolution\_clock::now();

std::cout << "STL Sort execution time: " << std::chrono::duration\_cast<std::chrono::microseconds>(end - start).count() << " microseconds" << std::endl;

std::cout << "First 10 elements: ";

for (int i = 0; i < 10; i++) {

std::cout << arr[i] << " ";

}

std::cout << std::endl;

std::cout << "Last 10 elements: ";

for (int i = 0; i < 10; i++) {

std::cout << arr[arr.size() - 1 - i] << " ";

}

std::cout << std::endl;

}

**OUTPUT:**

